PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the application of: **Lesartre *et al.***	)	Confirmation No.:	3656
Serial No.: 10/632,772	)	Art Unit:	2195
Filed: August 1, 2003	)	Examiner:	Wai, Eric Charles
For: PROCESSOR PURGING SYSTEM AND METHOD	)	Docket No.:	200209214-1

APPEAL BRIEF UNDER 37 C.F.R. §41.37

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief under 37 C.F.R. §41.37 is submitted in support of the Notice of Appeal filed January 24, 2008, responding to the final Office Action of September 25, 2007.

It is not believed that extensions of time or fees for net addition of claims are required, beyond those which may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. §1.136(a), and any fees required therefor (including fees for net addition of claims) are hereby authorized to be charged to Hewlett-Packard Company Deposit Account No. 08-2025.

I. REAL PARTY IN INTEREST

The real party in interest of the instant application is the assignee, Hewlett-Packard Development Company, L.P.

II. RELATED APPEALS AND INTERFERENCES

There are no known related appeals and interferences that will affect or be affected by a decision in this appeal.

III. STATUS OF THE CLAIMS

Claims 1, 3-8, 12-14, 16, 17, and 21-31 are pending in the present application. The final Office Action of September 25, 2007, rejected claims 1, 3, 4, 12, 16, 17, and 21-31 under 35 U.S.C. §103 as allegedly unpatentable over Applicants' Admitted Prior Art ("AAPA") in view of *Moore* (U.S. Patent No. 5,437,017). The final Office Action also rejected claims 5-8, 13, and 14 under 35 U.S.C. §103 as allegedly unpatentable over AAPA and *Moore* in view of *Mathews* (U.S. Patent No. 6,560,689). The final Office Action objected to the drawings and claims 3 and 5 as containing various alleged informalities. The final Office Action further rejected claim 27 under 35 U.S.C. §112, first paragraph, as allegedly based on a disclosure which is not enabling. It is believed that the objections to the drawings and claims 3 and 5, as well as the rejection of claim 27 under 35 U.S.C. §112, first paragraph, have been overcome via amendments submitted by Applicants on November 9, 2007. The final rejections of claims 1, 3-8, 12-14, 16, 17, and 21-31 are appealed.

IV. STATUS OF AMENDMENTS

Since the mailing of the final Office Action, Applicants submitted amendments on November 9, 2007, amending claims 3, 22, 27, and 30. In an Advisory Action mailed on November 30, 2007, it is indicated that the Patent Office will enter the amendments of November 9, 2007. A copy of the current claims, as amended via the amendments of November 9, 2007, is attached hereto as Appendix A.

V. SUMMARY OF CLAIMED SUBJECT MATTER

A processor purging system (*e.g.*, reference numeral 300) of some embodiments, such as that embodied by claim 1, comprises a translation lookaside buffer (TLB) (*e.g.*, reference numeral 306) having a plurality of translation pairs and at least one memory cache (*e.g.*, Paragraph [0028], lines 1-4). See Figures 3 and 4. The processor purging system also comprises logic (*e.g.*, reference numerals 308, 414, 302) configured to make a determination whether at least one of the translation pairs corresponds to a purge signal (*e.g.*, Paragraph [0034], lines 1-6, and Paragraph [0035], lines 1-13) and to purge, in response to the purge signal, each of the translation pairs in the TLB corresponding to the purge signal (*e.g.*, Paragraph [0030], lines 1-6). The logic is further configured to transmit, based on the determination, a purge detection signal indicative of whether at least one translation pair in the TLB corresponds to the purge signal (*e.g.*, Paragraph [0035], lines 9-13) and to determine, based upon the purge detection signal, whether to search the memory cache for information to be purged based on the purge signal (*e.g.*, Paragraph [0041], lines 1-7).

A method of some embodiments, such as that embodied by claim 12, is for purging a processor (*e.g.*, reference numeral 301). The method comprises the steps of detecting whether at least one of a plurality of translation pairs in a translation lookaside buffer (TLB) (*e.g.*, reference numeral 306) corresponds to a purge signal (*e.g.*, Paragraph [0034], lines 1-6, and Paragraph [0035], lines 1-13) and, if at least one of the translation pairs in the TLB corresponds to the purge signal, purging the at least one translation pair corresponding to the

purge signal (e.g., Paragraph [0030], lines 1-6). The method further comprises the steps of transmitting, based on the detecting step, a purge detection signal indicative of whether at least one of the translation pairs in the TLB corresponds to the purge signal (e.g., Paragraph [0035], lines 9-13) and determining whether to purge an instruction queue based on the purge detection signal (e.g., Paragraph [0041], lines 1-7).

A method of some embodiments, such as that embodied by claim 17, is for purging a processor (e.g., reference numeral 301). The method comprises detecting whether at least one translation pair in a plurality of translation pairs within a translation lookaside buffer (TLB) (e.g., reference numeral 306) corresponds to a purge signal (e.g., Paragraph [0034], lines 1-6, and Paragraph [0035], lines 1-13) and transmitting, based on the detecting step, a purge detection signal indicative of whether at least one of the translation pairs corresponds to the purge signal (e.g., Paragraph [0035], lines 9-13). The method further comprises determining, based upon the purge detection signal, whether to search the memory cache for information to be purged based on the purge signal (e.g., Paragraph [0041], lines 1-7) and, if at least one of the translation pairs in the TLB corresponds to the purge signal, purging from the TLB the at least one translation pair corresponding to the purge signal (Paragraph [0030], lines 1-6).

A processor (e.g., reference numeral 301) of some embodiments, such as that embodied by claim 25, comprises an execution unit (e.g., reference numeral 302) and an instruction queue (e.g., reference numeral 412) coupled to the execution unit (e.g., Figures 3 and 4). The processor also comprises a translation lookaside buffer (TLB) (e.g., reference numeral 306) configured to store a plurality of translation pairs, and each translation pair has a respective virtual address and a respective physical address (e.g. Paragraph [0019], lines 3-4, and Paragraph [0022], lines 4-7). The processor further comprises logic (e.g., reference numerals 308, 414, 302) configured to receive a purge signal and to make a determination as to whether any of the translation pairs stored in the TLB correspond to the purge signal (e.g., Paragraph [0034], lines 1-6, and Paragraph [0035], lines 1-13). The logic is further configured to purge from the TLB each translation pair corresponding to the purge signal

(*e.g.*, Paragraph [0030], lines 1-6) and to determine, based on the determination, whether to purge the instruction queue in response to the purge signal (*e.g.*, Paragraph [0041], lines 1-7).

A method of some embodiments, such as that embodied by claim 28, comprises the step of storing a plurality of translation pairs in a translation lookaside buffer (TLB) (*e.g.*, reference numeral 306), each of the translation pairs having a respective virtual address and a respective physical address (*e.g.* Paragraph [0019], lines 3-4, and Paragraph [0022], lines 4-7). See Figure 3. The method further comprises the steps of receiving a purge signal identifying at least one stale translation pair and determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal (*e.g.*, Paragraph [0034], lines 1-6, and Paragraph [0035], lines 1-13). The method also comprises the steps of, if at least one of the plurality of translation pairs in the TLB is identified by the purge signal, purging the at least one translation pair identified by the purge signal (*e.g.*, Paragraph [0030], lines 1-6) and determining whether to purge at least one component of a memory cache other than the TLB based on the step of determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal (*e.g.*, Paragraph [0041], lines 1-7).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1, 3, 4, 12, 16, 17, and 21-31 are rejected under 35 U.S.C. §103 as allegedly unpatentable over Applicants' Admitted Prior Art ("AAPA") in view of *Moore* (U.S. Patent No. 5,437,017).

Claims 5-8, 13, and 14 are rejected under 35 U.S.C. §103 as allegedly unpatentable over AAPA and *Moore* in view of *Mathews* (U.S. Patent No. 6,560,689).

VII. ARGUMENT

35 U.S.C. §103 Rejections

In order for a claim to be properly rejected under 35 U.S.C. §103, the combined teachings of the prior art references must suggest all features of the claimed invention to one of ordinary skill in the art. See, e.g., *In Re Dow Chemical Co.*, 837 F.2d 469, 5 U.S.P.Q.2d 1529, 1531 (Fed. Cir. 1988). In addition, "(t)he PTO has the burden under section 103 to establish a *prima facie* case of obviousness." *In re Fine*, 837 F.2d 1071, 5 U.S.P.Q.2d 1596, 1598 (Fed. Cir. 1988). "On the issue of obviousness, the combined teachings of the prior art as a whole must be considered." *EWP Corp. v. Reliance Universal Inc.*, 755 F.2d 898, 225 U.S.P.Q. 20, 25 (Fed. Cir. 1985).

Discussion of 35 U.S.C. §103 Rejections of Claims 1, 2-8, 12-14, 16, 17, and 21-24

Claim 1 presently stands rejected in the final Office Action under 35 U.S.C. §103 as allegedly unpatentable over Applicants' Admitted Prior Art ("AAPA") in view of *Moore* (U.S. Patent No. 5,437,017). Claims 12 and 17 comprise similar claimed limitations which are missing from the alleged combination (with respect to the outstanding 35 U.S.C. §103 rejections) as claim 1. Claims 2-8, 13, 14, 16, and 21-24 depend from a respective one of the independent claims 1, 12, or 17. Therefore, claim 1 is discussed below as an exemplary claim for discussion.

Claim 1 reads as follows:

> "1. A processor purging system, comprising:
> a translation lookaside buffer (TLB) having a plurality of translation pairs;
> at least one memory cache; and
> logic configured to make a determination whether at least one of the translation pairs corresponds to a purge signal and to purge, in response to the purge signal, each of the translation pairs in the TLB corresponding to the purge signal, ***the logic further configured to transmit, based on the determination, a purge detection signal indicative of whether at least one translation pair in the TLB corresponds to the purge signal and to determine, based upon the purge detection signal, whether to search the memory cache for information to be purged based on the purge signal.***"
> (Emphasis added).

Applicants respectfully assert that the alleged combination fails to suggest at least the features of claim 1 highlighted above. Thus, the 35 U.S.C. §103 rejection of claim 1 is improper.

In this regard, it is candidly admitted in the final Office Action that:

> "AAPA does not explicitly teach that the logic further configured to transmit, based on the determination, a purge detection signal indicative of whether at least one translation pair in the TLB corresponds to the purge signal and to determine, based upon the purge detection signal, whether to search the memory cache for a information to be purged based on the purge signal."

However, it is alleged in the final Office Action that *Moore* teaches the features missing from the AAPA and that it would be obvious to combine *Moore* with the AAPA. In particular, it is alleged that:

> "Moore teaches a system and method for maintaining TLB coherency between the TLB and memory queues (i.e. instruction queue) in a multiprocessor system (abstract). Moore accomplishes this by broadcasting a TLB invalidate instruction to all processors (Fig. 4). Once all processors have indicated that the instruction has completed execution, a determination is made whether the memory queue has reached coherency (col. 9 lines 30-42).
>
> It would have been obvious to one of ordinary skill in the art at the time of the inventions to include transmitting a signal to indicate when to check for coherency in the memory queue as taught by Moore. One would be motivated by the desire to ensure that the TLB was purged of all invalid entries before performing a coherency check on the rest of the memory queues."

Applicants respectfully assert that, even if it is assumed for the sake of argument that the above allegations are true, the final Office Action nevertheless fails to establish a *prima facie* case of obviousness, and the 35 U.S.C. §103 rejection of claim 1 is, therefore, improper for at least this reason.

In this regard, *Moore* apparently teaches that a translation lookaside buffer invalidate (TLBI) instruction is broadcast to each processor in a multiprocessor system. See Abstract and column 8, lines 37-44. If all of the processors "accept" the TLBI instruction, then each processor apparently initiates a purge of its respective translation lookaside buffer (TLB). However, if any of the processors do not accept the TLBI instruction, then none of the processors initiate execution of the TLBI instruction. In this way, coherency among the TLBs of the multiprocessor system can be maintained. See Abstract and column 8, line 45,

to column 9, line 8. If all of the processors accept the TLBI instruction, then execution of pending instructions is temporarily terminated, and once it can be ensured that no pending instructions are about to execute, the TLBI instruction is executed in each processor. See Abstract and column 9, lines 9-29. Thus, the TLB for each processor is apparently purged based on the TLBI instruction. Once a processor's TLB is purged, the processor's memory queue 36 is checked to determine whether it has achieved coherency. See column 9, lines 30-41.

However, there is nothing in *Moore* to indicate that any "purge detection signal" is used in order to check the memory queue 36 for coherency. In this regard, a "purge detection signal," as recited by claim 1, is a signal that indicates "whether at least one translation pair in the TLB corresponds to the purge signal." Indeed, it appears that the memory queue 36 in *Moore* is checked for coherency ***regardless*** of whether any of the translation pairs in the TLB correspond to the alleged "purge signal" (see Figure 5, blocks 122 and 124), and there is nothing in *Moore* to suggest that the searching of the memory queue 36 should be affected in any way based on a determination as to whether any of the translation pairs in the TLB correspond to the "purge signal." Accordingly, the Office Action fails to establish a *prima facie* case of obviousness with respect to at least the features of "the logic further configured to transmit, based on the determination, a purge detection signal indicative of whether at least one translation pair in the TLB corresponds to the purge signal and to determine, ***based upon the purge detection signal***, whether to search the memory cache for information to be purged based on the purge signal," as recited by claim 1. (Emphasis added).

In the Advisory Action mailed on November 30, 2007, it is indicated that the Examiner agrees with Applicants' assertions that the memory queue 36 in *Moore* is checked for coherency regardless of whether any of the translation pairs in the TLB correspond to the alleged "purge signal." However, it is alleged in the Advisory Action that the AAPA, at Paragraph [0005] of the instant application, teaches that "the determination to purge instruction queues is performed based on the processor's TLB containing a translation pair

related to the modification in the page table." Applicants respectfully disagree. In this regard, the sentence in the AAPA at issue reads as follows:

> "In this regard, if a system change occurs that modifies the page table, then the operating system transmits a purge signal to each of the processors in the multiprocessor system, so that ***if a processor's TLB contains a translation pair related to the modification, then the processor can purge the TLB translation pair and purge any mini-TLBs or instruction queues that may attempt to use data related to the deleted address***." (Emphasis added).

The sentence indicates that two events occur if a processor's TLB contains a translation pair related to the modification: (1) the processor purges the TLB pair from the TLB and (2) the processor purges any mini-TLBs or instructions queues that may attempt to use data related to the deleted address. This is certainly true in that both events do, in fact, occur when there is a TLB translation pair that corresponds to a "purge signal." However, the sentence is silent on what happens if the processor's TBL does ***not*** contain a translation pair related to the modification. Indeed, the sentence fails to indicate or suggest that mini-TLBs or instruction queues are not checked or purged if there is a miss in the TLB.

In addition, when Applicants' remarks at Paragraph [0005] are properly viewed as a whole in the context of other remarks made by Applicants in the instant application, it is clear that Applicants do not admit that, in the prior art, purging or checking mini-TLBs or instruction queues are based on whether there is a TLB hit. "It is impermissible within the framework of section 103 to pick and choose from any one reference only so much of it as will support a given position, to the exclusion of other parts necessary to the full appreciation of what such reference fairly suggests to one of ordinary skill in the art." *In re Wesslau*, 353 F.2d 238, 147 U.S.P.Q. 391, 393 (C.C.P.A. 1965). In this regard, a clear description of how a conventional system responds to a "purge signal" is described at Paragraph [0023], which reads:

> "During operation of system 100, the operating system 120 periodically transmits "purge signals" to the processor 102 in response to system events that indicate one or more translation pairs may be stale, e.g., the virtual address or PA of the page may have changed. Such purge signals may indicate that the status of the page has changed, and any reference to such page needs to be updated in the TLB. ***In response to such a purge signal, the MMU 108 searches the TLB 112 for the stale translation pair identified by the purge signal and purges the stale translation pair, if it is present in***

the TLB 112. The MMU 108 also purges other processor components, for example mini-TLBs or instruction queues that may be affected by the stale translation pair. Moreover, the processor 102 may purge the L1 cache 110 and/or any instruction queue that may be resident on the processor 102 and that could possibly contain a page reference to the purged translation pair. In so purging, the processor 102 eliminates the risk that it will attempt to access stale or unavailable data via a stale translation pair or that it will attempt to execute an instruction related to the stale translation pair. ***Typically, these other structures are flushed on the receipt of any TLB purge signal.***" (Emphasis added).

Moreover, it is clear that the admitted prior art does ***not*** include a recognition that it is well known to purge mini-TLBs or instruction queues based on a "purge detection signal," which is "indicative of whether at least one translation pair in the TLB corresponds to the purge signal," as recited by claim 1.

For at least the above reasons, Applicants assert that the cited art fails to suggest each feature of claim 1 and the final Office Action fails to establish a *prima facie* case of obviousness with respect to claim 1. Accordingly, Applicants respectfully request that the 35 U.S.C. §103 rejection of claim 1 be overruled.

Discussion of 35 U.S.C. §103 Rejections of Claims 25-27

Claim 25 presently stands rejected in the final Office Action under 35 U.S.C. §103 as allegedly unpatentable over AAPA in view of *Moore*. Claims 26 and 27 depend from claim 25. Therefore, claim 25 is discussed below as an exemplary claim for discussion.

Claim 25 reads as follows:

"25. A processor, comprising:
an execution unit;
an instruction queue coupled to the execution unit;
a translation lookaside buffer (TLB) configured to store a plurality of translation pairs, each translation pair having a respective virtual address and a respective physical address; and
logic configured to receive a purge signal and to make a determination as to whether any of the translation pairs stored in the TLB correspond to the purge signal, the logic configured to purge from the TLB each translation pair corresponding to the purge signal, the logic further configured to determine, based on the determination, whether to purge the instruction queue in response to the purge signal." (Emphasis added).

Applicants respectfully assert that the cited art fails to suggest at least the features of claim 25 highlighted above.

As set forth above in the Discussion of 35 U.S.C. §103 Rejections of Claims 1, 2-8, 12-14, 16, 17, and 21-24, *Moore* apparently teaches that the TLB for each processor in a multiprocessor system is purged based on a TLBI instruction broadcast to each processor, provided that each processor accepts the instruction. For each processor, the memory queue 36 of *Moore* is checked to determine whether it has achieved coherency after the processor's TLB has been purged. See column 9, lines 30-41. However, the memory queue 36 is apparently checked for coherency ***regardless*** of whether any of the translation pairs in the TLB correspond to the alleged "purge signal" (see Figure 5, blocks 122 and 124), and there is nothing in *Moore* to suggest that the searching of the memory queue 36 should be affected in any way based on a determination as to whether any of the translation pairs stored in the TLB correspond to the "purge signal." Accordingly, the final Office Action fails to establish a *prima facie* case of obviousness with respect to at least the features of "logic configured to receive a purge signal and to make a determination as to whether any of the translation pairs stored in the TLB correspond to the purge signal, the logic configured to purge from the TLB each translation pair corresponding to the purge signal, ***the logic further configured to determine, based on the determination, whether to purge the instruction queue in response to the purge signal***," as recited by claim 25. (Emphasis added). In addition, for at least reasons similar to those set forth above in the Discussion of 35 U.S.C. §103 Rejections of Claims 1, 2-8, 12-14, 16, 17, and 21-24, Applicants respectfully assert that the admitted prior art does ***not*** include a recognition that it is well known to purge mini-TLBs or instruction queues based on a "determination as to whether any of the translation pairs stored in the TLB correspond to the purge signal," as recited by claim 25.

For at least the above reasons, Applicants assert that the cited art fails to suggest each feature of claim 25 and the final Office Action fails to establish a *prima facie* case of

obviousness with respect to claim 25. Accordingly, Applicants respectfully request that the 35 U.S.C. §103 rejection of claim 25 be overruled.

Discussion of 35 U.S.C. §103 Rejections of Claims 28-31

Claim 28 presently stands rejected in the final Office Action under 35 U.S.C. §103 as allegedly unpatentable over AAPA in view of *Moore*. Claims 29-31 depend from claim 28. Therefore, claim 28 is discussed below as an exemplary claim for discussion.

> 28. A method, comprising the steps of:
> storing a plurality of translation pairs in a translation lookaside buffer (TLB), each of the translation pairs having a respective virtual address and a respective physical address;
> receiving a purge signal identifying at least one stale translation pair;
> ***determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal;***
> if at least one of the plurality of translation pairs in the TLB is identified by the purge signal, purging the at least one translation pair identified by the purge signal; and
> ***determining whether to purge at least one component of a memory cache other than the TLB based on the step of determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal.*** (Emphasis added).

Applicants respectfully assert that the cited art fails to suggest at least the features of claim 28 highlighted above.

As set forth above in the Discussion of 35 U.S.C. §103 Rejections of Claims 1, 2-8, 12-14, 16, 17, and 21-24, *Moore* apparently teaches that the TLB for each processor in a multiprocessor system is purged based on a TLBI instruction broadcast to each processor, provided that each processor accepts the instruction. For each processor, the memory queue 36 of *Moore* is checked to determine whether it has achieved coherency after the processor's TLB has been purged. See column 9, lines 30-41. However, the memory queue 36 is apparently checked for coherency ***regardless*** of whether any of the translation pairs in the TLB are identified by the alleged "purge signal" (see Figure 5, blocks 122 and 124), and there is nothing in *Moore* to suggest that the searching of the memory queue 36 should be affected in any way based on a determination as to whether any of the translation pairs stored in the

TLB are identified by the "purge signal." Accordingly, the final Office Action fails to establish a *prima facie* case of obviousness with respect to at least the features of "determining whether to purge at least one component of a memory cache other than the TLB based on the step of determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal," as recited by claim 28. In addition, for at least reasons similar to those set forth above in the Discussion of 35 U.S.C. §103 Rejections of Claims 1, 2-8, 12-14, 16, 17, and 21-24, Applicants respectfully assert that the admitted prior art does ***not*** include a recognition that it is well known to purge mini-TLBs or instruction queues based on the step of "determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal," as recited by claim 28.

For at least the above reasons, Applicants assert that the cited art fails to suggest each feature of claim 28 and the final Office Action fails to establish a *prima facie* case of obviousness with respect to claim 28. Accordingly, Applicants respectfully request that the 35 U.S.C. §103 rejection of claim 28 be overruled.

CONCLUSION

Based on the foregoing discussion, Applicant respectfully requests that the Examiner's final rejections of claims 1, 3-8, 12-14, 16, 17, and 21-31 be overruled and withdrawn by the Board, and that the application be allowed to issue as a patent with all pending claims.

Respectfully submitted,

THOMAS, KAYDEN, HORSTEMEYER & RISLEY, L.L.P.

By: 

Jon E. Holland
Reg. No. 41,077
(256) 704-3900 Ext. 103

Hewlett-Packard Company
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

VIII. CLAIMS - APPENDIX

1. A processor purging system, comprising:

a translation lookaside buffer (TLB) having a plurality of translation pairs;

at least one memory cache; and

logic configured to make a determination whether at least one of the translation pairs corresponds to a purge signal and to purge, in response to the purge signal, each of the translation pairs in the TLB corresponding to the purge signal, the logic further configured to transmit, based on the determination, a purge detection signal indicative of whether at least one translation pair in the TLB corresponds to the purge signal and to determine, based upon the purge detection signal, whether to search the memory cache for information to be purged based on the purge signal.

2. (Canceled)

3. The system of claim 1, wherein the memory cache further comprises an instruction queue.

4. The system of claim 3, wherein the memory cache further comprises a mini-TLB.

5. The system of claim 1, further wherein the logic is configured to compare the purge signal with each of the plurality of translation pairs and to transmit a plurality of match signals corresponding respectively to the plurality of translation pairs, each of the match signals indicating whether the corresponding translation pair corresponds to the purge signal.

6. The system of claim 5, wherein the logic is further configured to collapse the match signals into the purge detection signal.

7. The system of claim 6, wherein the logic comprises a plurality of tiered logical AND gates configured to collapse the match signals into the purge detection signal.

8. The system of claim 6, wherein the logic comprises a plurality of tiered logical OR gates configured to collapse the match signals into the purge detection signal.

9-11. (Canceled)

12. A method for purging a processor, comprising the steps of:

detecting whether at least one of a plurality of translation pairs in a translation lookaside buffer (TLB) corresponds to a purge signal;

if at least one of the translation pairs in the TLB corresponds to the purge signal, purging the at least one translation pair corresponding to the purge signal;

transmitting, based on the detecting step, a purge detection signal indicative of whether at least one of the translation pairs in the TLB corresponds to the purge signal; and

determining whether to purge an instruction queue based on the purge detection signal.

13. The method of claim 12, wherein the detecting step further comprises the steps of:

comparing the purge signal with each of the translation pairs; and

transmitting match signals, each of the match signals indicative of whether a respective one of the translation pairs corresponds to the purge signal.

14. The method of claim 13, further comprising the step of collapsing each of the match signals into the purge detection signal.

15. (Canceled)

16. The method of claim 12, further comprising the step of purging, based on the determining step, the instruction queue if the purge detection signal indicates that at least one translation pair in the TLB corresponds to the purge signal.

17. A processor purging method, comprising:

detecting whether at least one translation pair in a plurality of translation pairs within a translation lookaside buffer (TLB) corresponds to a purge signal;

transmitting, based on the detecting step, a purge detection signal indicative of whether at least one of the translation pairs corresponds to the purge signal;

determining, based upon the purge detection signal, whether to search the memory cache for information to be purged based on the purge signal; and

if at least one of the translation pairs in the TLB corresponds to the purge signal, purging from the TLB the at least one translation pair corresponding to the purge signal.

18-20. (Canceled)

21. The system of claim 3, wherein the logic is further configured to determine whether to purge the instruction queue based on the purge detection signal.

22. The method of claim 12, further comprising the step of purging the instruction queue if the purge detection signal indicates that at least one of the translation pairs in the TLB corresponds to the purge signal.

23. The method of claim 12, wherein the determining step comprises the step of determining not to purge the instruction queue in response to the purge signal if none of the translation pairs correspond to the purge signal.

24. The method of claim 17, further comprising the step of purging an instruction queue in response to the purge signal if a detection is made in the detecting step that at least one of the translation pairs within the TLB corresponds to the purge signal.

25. A processor, comprising:

an execution unit;

an instruction queue coupled to the execution unit;

a translation lookaside buffer (TLB) configured to store a plurality of translation pairs, each translation pair having a respective virtual address and a respective physical address; and

logic configured to receive a purge signal and to make a determination as to whether any of the translation pairs stored in the TLB correspond to the purge signal, the logic configured to purge from the TLB each translation pair corresponding to the purge signal, the logic further configured to determine, based on the determination, whether to purge the instruction queue in response to the purge signal.

26. The processor of claim 25, wherein the logic is configured to purge the instruction queue in response to the purge signal if any of the translation pairs stored in the TLB correspond to the purge signal.

27. The processor of claim 26, wherein the logic is configured to refrain from purging the instruction queue in response to the purge signal unless at least one of the translation pairs stored in the TLB corresponds to the purge signal.

28. A method, comprising the steps of:

storing a plurality of translation pairs in a translation lookaside buffer (TLB), each of the translation pairs having a respective virtual address and a respective physical address;

receiving a purge signal identifying at least one stale translation pair;

determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal;

if at least one of the plurality of translation pairs in the TLB is identified by the purge signal, purging the at least one translation pair identified by the purge signal; and

determining whether to purge at least one component of a memory cache other than the TLB based on the step of determining whether any of the plurality of translation pairs in the TLB are identified by the purge signal.

29. The method of claim 28, wherein the step of determining whether to purge the at least one component step comprises the step of determining not to purge the at least one component if none of the translation pairs in the TLB is identified by the purge signal.

30. The method of claim 28, further comprising the step of, if at least one of the plurality of translation pairs in the TLB is identified by the purge signal, purging the at least one component in response to the step of determining whether to purge the at least one component.

31. The method of claim 30, wherein the at least one component comprises an instruction queue.

IX. EVIDENCE - APPENDIX

None.

X. RELATED PROCEEDINGS - APPENDIX

None.